MULTIMEDIA UNIVERSITY

TABLE NO: | | | |

STUDENT ID NO: | | | | | | | | | | |

VENUE: _____

MULTIMEDIA UNIVERSITY

FINAL EXAMINATION

TRIMESTER 2, 2019/2020

TCP1101 – PROGRAMMING FUNDAMENTALS

(All sections / Groups)

9 MARCH 2020
2.30 p.m. – 4.30 p.m.
(2 Hours)

Question	Mark
1	
2	
3	
4	
Total:	

Examiner 1 Signature: _____

Examiner 2 Signature: _____

INSTRUCTIONS TO STUDENT

1. This Question paper consists of **15** printed pages (including cover page) with 4 Questions only.

2. Answer **ALL QUESTIONS.** All questions carry equal marks. The distribution of the marks for each question is given.

3. Please print all your answers **CLEARLY** in the specific answer boxes provided for each question. Submit this question paper at the end of the examination.

Unless otherwise stated, every code segment provided in every question is contained within a main function, complete with the directive **`#include <iostream>`** and the statements **`using namespace std;`** and **`return 0;`**

QUESTION 1 [10 Marks]

a) A prime number is an integer greater than one that is only divisible evenly by 1 and itself.

The program in the **Code** box intends to take two positive integers A and B (A is always less than B, and A, B > 1), and identify all prime numbers between A and B inclusively. It outputs the identified prime numbers in ascending order and the number of prime numbers identified. However, there are missing codes in the program. Each of them is indicated by an underlined block comment (/* */).

Fill in the missing codes by choosing the most appropriate answer for each question in the **Questions** box. Write your answer (i.e. A, B, etc.) in the Answer box (i.e. the right-most column for each question). Refer to the sample runs below for the intended sample output (user input is underlined).

[3 marks]

Sample Run 1:

```
Enter A and B: 23 47
Primes: 23 29 31 37 41 43 47
There are 7 prime(s) between 23 and 47
```

Sample Run 2:

```
Enter A and B: 2 10
Primes: 2 3 5 7
There are 4 prime(s) between 2 and 10
```

Code:

```
#include <iostream>
#include <cmath>
using namespace std;

/* Q1: isPrime() function header */
{
  for (int i = 2; /* Q2: for-loop condition */ ; ++i)
    if (num % i == 0)
    {
      /* Q3: return statement */
    }

  /* Q4: return statement */
}

int main()
{
  int A, B;
  cout << "Enter A and B: ";
  cin >> A >> B;
```

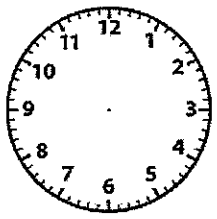
```
  cout << "Primes: ";
  int countPrime = 0;
  for (int i = A; i <= B; ++i)
    if ( /* Q5: isPrime function call */ )
    {
      /* Q6: Update countPrime */
      cout << i << " ";
    }

  cout << "\nThere are " << countPrime
       << " prime(s) between "
       << A << " and " << B << endl;

  return 0;
}
```

Questions: **Answer**

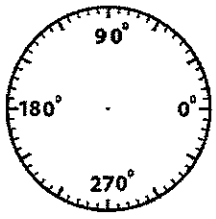
Q1	(A) bool isPrime() (B) bool isPrime(int num) (C) void isPrime() (D) void isPrime(int num)	
Q2	(A) i <= num (B) i < num (C) All of the above are correct.	
Q3	(A) return true; (B) return false; (C) return static_cast<bool>(1); (D) return static_cast<bool>(0); (E) Only (A) and (C) are correct. (F) Only (B) and (D) are correct.	
Q4	(A) return true; (B) return false; (C) return static_cast<bool>(1); (D) return static_cast<bool>(0); (E) Only (A) and (C) are correct. (F) Only (B) and (D) are correct.	
Q5	(A) isPrime() (B) isPrime() == true (C) isPrime(i) (D) isPrime(i) == true (E) Only (A) and (B) are correct. (F) Only (C) and (D) are correct.	
Q6	(A) countPrime = countPrime + 1; (B) ++countPrime; (C) countPrime++; (D) Only (B) and (C) are correct. (E) All of the above are correct.	

Continued ...

b) Figure below shows an analogue clock and a degree wheel.



Analogue clock



Degree Wheel

The program in the **Code** block intends to convert hour in the analogue clock to degree in the degree wheel. However, the program has missing function definition.

Complete the function definition so that the program works as intended. To get full marks, your answers **must** fulfill all the following requirements:

1. **Do not** use array and any classes from C++ Standard Template Library.
2. **Do not** use *else if* statement and nested *if/else* statements (i.e. at most one *if* statement and one *else* statement are allowed).

Note: Partial marks will be given if your answer is correct but violates any of the requirements above.

Write your answer in the **Answer** boxes given. Refer to the sample runs below for the intended sample output (user input is underlined).

[5 marks]

Sample Run 1:

```
Enter minute: 15
15 minute => 0 degree
```

Sample Run 2:

```
Enter minute: 16
16 minute => 354 degree
```

Sample Run 3:

```
Enter minute: 30
30 minute => 270 degree
```

Code:

```
#include <iostream>
using namespace std;

/* MinuteToDegree() function definition */

int main()
{
   cout << "Enter minute: ";
   int minute;
   cin >> minute;
```

```
    int mDegree = MinuteToDegree(minute);

    cout << minute << " minute => "
         << mDegree << " degree" << endl;

    return 0;
}
```

Answers:

```
/* MinuteToDegree() function definition */
```

Continued ...

c) Write the output as the result of execution of the following code segments.

[2 marks]

Q1	`int x = 10;`
	`do`
	`{`
	`  x = x - 3;`
	`}`
	`while(x > 0);`
	`cout << x << endl;`	

Q2	`int op = 1;`
	`int a = 0;`
	
	`switch(op)`
	`{`
	`  case 0:  a += (3 + 4);`
	`  case 1:  a += (3 - 4);`
	`  case 2:  a += (3 * 4);`
	`  default: a += (3 / 4);`
	`}`
	
	`cout << a << endl;`	

Q3	`void swapInt(int a, int b)`
	`{`
	`  int temp = a;`
	`  a = b;`
	`  b = temp;`
	`}`
	
	`int main()`
	`{`
	`  int a = 11, b = 99;`
	`  swapInt(a, b);`
	`  cout << a << "," << b << endl;`
	`}`	

Q4	`char c[] = {'m', 'i', 'a'};`
	`int d[4] = {1, 0, 2};`
	`for (int i = 0; i < 4; ++i)`
	`  cout << c[ d[i] ];`	

QUESTION 2 [10 Marks]

a) Define a function named `arrangeZero` that accepts two parameters: (1) `A`, an integer array (2) `size`, the capacity of array `A`. The function returns an integer indicating the number of zeroes in A.

The function also reorders the integers in `A` such that:

1. Non-zero integers are shifted to the left, while preserving the original order.
2. Zeroes are shifted to the right, after the non-zeroes.

You may or may not use the following algorithm to reorder the integers:

1. Create a resizable array `B` to store the locations of non-zeroes in `A`.
2. Populate `A` with non-zeroes on the left with the help of `B`.
3. Populate `A` with zeroes on its right, after the last non-zero integer.

Write your answer in the **Answer** box given. Check out the example below on how the function works.

[6 marks]

```
int arr[] = {0, 1, 0, 2, -3, 0, 4};
int count = arrangeZero(arr, 7);

// At this point, count should be 3, and
// arr should be {1, 2, -3, 4, 0, 0, 0}
```

Answer:

```
#include <vector>
/* arrangeZero() function definition */
```

Continued ...

b) A quiz contains several multiple-choice questions, each of which has 4 choices: A, B, C, and D.

The program in the **Code** box intends to evaluate a student's quiz answers. First, it reads the quiz's true answers from a text file named "ans.txt" into memory. Then, it asks for a student's quiz answers. Finally, it compares the student's answers with the true answers, and outputs the percentage of questions the student has answered correctly, **rounded to two decimal places**. Table below shows sample runs for some given input (underlined) and content of the ans.txt text file.

No.	Sample Runs	Content of ans.txt
1	Enter student's answers: A D C Percentage: 66.67%	A B C
2	Enter student's answers: B B B B B Percentage: 20.00%	D C D D B
3	Enter student's answers: A A A A A C C C C Percentage: 44.44%	A B A D D C C B D

Unfortunately, the program has missing codes. Each of them is indicated by an underlined block comment (/* */). Fill in the missing codes by choosing the most appropriate answer for each question in the **Questions** box. Write your answer (i.e. A, B, etc.) in the **Answer** box (i.e. the right-most column for each question).

[4 marks]

Code:

```
#include <iostream>
#include <fstream>
#include <vector>
#include <iomanip>
using namespace std;

void ReadTrueAnswers( /* Q1: Parameter(s) */ )
{
  char ans;
  ifstream fin("ans.txt");
  while ( /* Q2: Read one answer from file */ )
    answers.push_back(ans);
  fin.close();
}

float Compare( /* Q3: Data type of ansTrue */     ans1,
               /* Q4: Data type of ansStudent */ ans2)
{
  float percent = 0.0f;
  for (int i = 0; i < ans1.size(); ++i)
    if ( /* Q5: Compare answers from ans1 & ans2 */ )
      percent += 1.0f;
```

```
  return /* Q6: Return percentage of correct answers */ ;
}

int main()
{
  vector<char> ansTrue;
  ReadTrueAnswers(ansTrue);

  vector<char> ansStudent( /* Q7: Argument(s)? */ );
  cout << "Enter student's answers: " << endl;
  for (int i = 0; i < ansStudent.size(); ++i)
    cin >> answers[i];


  float percent = Compare(ansTrue, ansStudent);
  cout << "Percentage: "
       << /* Q8: IO Manipulator(s)? */ << percent << endl;
}
```

Questions:		**Answer**
Q1	(A) vector<char> answers (B) vector<char>& answers (C) All the above are correct.	
Q2	(A) fin >> ans (B) getline(fin, ans) (C) All the above are correct.	
Q3	(A) vector<char> (B) vector<char>& (C) const vector<char>& (D) All the above are correct.	
Q4	(A) vector<char> (B) vector<char>& (C) const vector<char>& (D) All the above are correct.	
Q5	(A) ans1 = ans2 (B) ans1[i] = ans2[i] (C) None of the above is correct. (D) All of the above are correct.	
Q6	(A) percent / 100.0f * ans1.size() (B) percent * 100.0f / ans1.size()	
Q7	(A) sizeof(ansTrue) (B) ansTrue.size() (C) None of the above is correct.	
Q8	(A) setprecision(2) (B) fixed << setprecision(2) (C) All of the above are correct.	

Continued ...

QUESTION 3 [10 Marks]

(a) Implement the class `PersonalInfo` and the function `displayPersonalInfo` so that the program displays the following output.

Program output:

```
Name: NoName
Address: NoAddress
Age: 0
Phone: NoPhone

Name: Ah Kau
Address: 48 Friendly Street
Age: 50
Phone: 555-8484

Name: Muthu
Address: 191 Apple Mountain Road
Age: 30
Phone: 555-2222
```

[6 marks]

```
#include <iostream>
#include <string>
using namespace std;

class PersonalInfo
{
   private:
      string name;
      string address;
      int age;
      string phone;

   public:
      PersonalInfo()
      {
         // Write your code here









      }

      PersonalInfo(string n, string addr, int a, string p)
      {
         // Write your code here









      }
```

```
        void setName(string n)
        { name = n; }

        void setAddress(string a)
        { address = a; }

        void setAge(int a)
        { age = a; }

        void setPhone(string p)
        { phone = p; }

        string getName() const
        { return name; }

        string getAddress() const
        { return address; }

        int getAge() const
        { return age; }

        string getPhone() const
        { return phone; }
};

void displayPersonalInfo( PersonalInfo obj )
{
   // Write your code here









}

int main()
{
   PersonalInfo me;
   PersonalInfo friend1("Ah Kau", "48 Friendly Street",
                        50, "555-8484");
   PersonalInfo friend2("Muthu", "191 Apple Mountain
                        Road", 30, "555-2222");

   displayPersonalInfo( me );
   displayPersonalInfo( friend1 );
   displayPersonalInfo( friend2 );

   return 0;
}
```

Continued ...

(b) Implement the function `stringLength`. The `stringLength` function accepts a pointer to a C-string and returns the number of characters in the C-string. Note that the function `strlen()` cannot be used because the library for `strlen()` is not included.

Sample output:

```
Enter a string of 80 characters or less:
Thank you!

The length of that string is: 10
```

[4 marks]

```
#include <iostream>
using namespace std;

int stringLength(char *str)
{
   // Write your code here









}

int main()
{
   const int SIZE = 81;
   char cstring[SIZE];

   // Get a string from the user.
   cout << "Enter a string of 80 characters " << "or less: \n";
   cin.getline(cstring, SIZE);

   cout << "\nThe length of that string is: "
        << stringLength(cstring) << endl;

   return 0;
}
```

Continued ...

QUESTION 4 [10 Marks]

(a) Write a program which can dynamically allocate an integer array to store integers depending on the number of integers required by users, and free up the allocated memory before the program ends. (*NOTE: C++ STL Vector is not allowed*).

Sample output:

```
Enter the number of integers the array has to store : 5
Enter integer 1 : 2
Enter integer 2 : 3
Enter integer 3 : 4
Enter integer 4 : 5
Enter integer 5 : 6
Echoing the array
2 3 4 5 6
```

[5 marks]

```
#include <iostream>
using namespace std;

int main( )
{
    // Write you code here









    return 0;
}
```

Continued ...

(b) Read the following program and fill in the blanks in the **Program Output**.

[5 marks]

```
#include<iostream>
#include<fstream>
using namespace std;

int main()
{
  long offset;
  long numBytes;     // To hold the file size
  char ch;
  char again;        // To hold Y or N

  // Open the file for input.
  fstream file("letters.txt", ios::in);

  // Determine the number of bytes in the file.
  file.seekg(0L, ios::end);
  numBytes = file.tellg();
  cout << "The file has " << numBytes << " bytes.\n";

  // Go back to the beginning of the file.
  file.seekg(0L, ios::beg);

  do
  {
    cout << "Currently at position " << file.tellg() << endl;
    cout << "Enter an offset from the file's current position: ";
    cin >> offset;

    if (offset >= numBytes)
      cout << "Cannot read past the end of the file.\n";
    else
    {
      file.seekg(offset, ios::cur);
      file.get(ch);
      cout << "Character read: " << ch << endl;
    }

    cout << "Do it again? ";
    cin >> again;
  } while (again == 'Y' || again == 'y');

  file.close();
  return 0;
}
```

Contents of letters.txt:

```
abcdefghijklmnopqrstuvwxyz
```

Continued ...

Program Output:

```
The file has 26 bytes.
Currently at position 0
Enter an offset from the current position of the file: 4
Character read:
Do it again? y
Currently at position
Enter an offset from the current position of the file: 3
Character read:
Do it again? y
Currently at position 9
Enter an offset from the current position of the file:
Character read: i
Do it again? n
```

End of Page